UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/731,597	12/09/2003	Robert Little	60001.0270US01/MS303919.1	4741

7590 08/09/2007

Christopher J. Leonard
Merchant & Gould P.C.
P.O. Box 2903
Minneapolis, MN 55402-0903

EXAMINER	
RUTLEDGE, AMELIA L	
ART UNIT	**PAPER NUMBER**
2176	

MAIL DATE	DELIVERY MODE
08/09/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/731,597	LITTLE ET AL.	
	Examiner	Art Unit	
	Amelia Rutledge	2176	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *16 May 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *7-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *7-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *5/16/07*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to communications: Amendment, filed 05/16/2007; IDS filed 05/16/2007; RCE filed 05/16/2007.

2. Claims 7-24 are pending in the case. Claims 7, 19, and 23 are independent claims.

3. Regarding applicant's summary of the interview held 02/26/2007, applicant discusses Figs. 13 and 14 of Skonnard, however, upon review of Skonnard, p. 12, no Fig. 14 is listed in the reference.

4. Claims 7-18 have been amended to overcome the previous rejections under 35 U.S.C. 101, therefore the claim rejections under 35 U.S.C. 101 have been withdrawn.

Requirement for Information under 37 CFR 1.105

1. Applicant and the assignee of this application are required under 37 CFR 1.105 to provide the following information that the examiner has determined is reasonably necessary to the examination of this application.

2. An issue of public use or on sale activity has been raised in this application because it is a continuation of the parent application. In order for the examiner to properly consider patentability of the claimed invention under 35 U.S.C. 102(b), additional information regarding this issue is required as follows:

3. The information disclosure statement filed 06/13/2006 cites a beta release with system code, however, a copy of the information has not been furnished with the IDS.

4. The information disclosure statement cites a beta release of Microsoft Office software which occurred in October 2002. Please provide copies of all documentation supplied with the beta release and/or related to the release, such as web pages and product manuals. A copy of the beta release is required. There is a requirement for information describing the method of distribution of the release, including a list of recipients of the beta release and a description of the method of selection of recipients.

5. Applicant is reminded that failure to fully reply to this requirement for information will result in a holding of abandonment. The applicant is reminded that the reply to this requirement must be made with candor and good faith under 37 CFR 1.56. Where the applicant does not have or cannot readily obtain an item of required information, a statement that the item is unknown or cannot be readily obtained may be accepted as a complete reply to the requirement for that item.

6. This requirement is an attachment of the enclosed Office action. A complete reply to the enclosed Office action must include a complete reply to this requirement. The time period for reply to this requirement coincides with the time period for reply to the enclosed Office action.

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 7-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Skonnard, "What's New in MSXML 4.0", p. 1-12, published December, 2001 in MSDN Magazine, in view of Armstrong, et al. (hereinafter "Armstrong"), "The Java Web Services Tutorial", published February 4, 2002 by Sun Microsystems, Inc., p. 1-84 and 329-366.

Regarding independent claim 7, which cites a *method for modifying the resources of a Markup Language (ML) schema library,* Skonnard teaches a Schema Object Model (SOM) which is loaded into a programmable object and associated with a namespace URI (p. 3, par. 1);. Skonnard teaches an API interface for allowing a user to select resources in a ML schema library (p. 4, par. 1; p. 2-3, "A New XML Schema API: SOM").

While Skonnard teaches a method to access a schema library, and calling and passing the root and node schema object properties of the schema library (p. 5-6), Skonnard does not explicitly teach *modifying the resources of a Markup Language (ML) schema library, comprising: calling the schema library via an object-oriented message call, wherein the object-oriented message call is configured to modify a ML schema file of the ML schema library;* since Skonnard teaches accessing properties of the schema library and that the user can drill down into a particular schema item using the appropriate interfaces (p. 6, par. 1-3), but does not explicitly teach modifying the schema file. However, Armstrong teaches several interfaces for passing object properties between Java, an object-oriented programming language, and XML.

Armstrong discloses the Java API for XML Processing, (JAXP), an object oriented API to modify an XML schema file (p. 6-8). Armstrong also teaches JAX_RPC, the Java API for XML-based RPC, which makes it possible for an application to define its own XML schema and to use that schema to send XML documents and XML fragments, as well as using the XML Simple Object Access Protocol (SOAP) to make an object oriented message call (p. 16-17). Since Armstrong teaches passing object properties using SOAP, and RPC (p. 16-17, p. 329-366), for web components (p. 337-338), i.e., software objects associated with functionality identified in the schema, Armstrong teaches *passing an object property, associated with the object-oriented message call, to the ML schema library, wherein the object property is associated with a software object associated with functionality identified in the ML schema library.*

Armstrong teaches, *in response to the message call and the object property passed to the ML schema library, modifying the functionality of the ML schema file identified in the ML schema library; and associating the modified functionality of the ML schema file with a document to govern the application of ML elements on the document*, because Armstrong teaches JAXR (p. 329-366) to access, submit data to an XML registry, modify it, and remove it (p. 355-360).

Both Skonnard and Armstrong are directed toward API interfaces for linking object-oriented programming languages with XML schema. It would have been obvious and desirable to one of ordinary skill in the art at the time of the invention to combine the methods of accessing XML schema libraries using an object-oriented schema interface (SOM) disclosed by Skonnard (Skonnard, p. 3, par. 2-4) with the flexible APIs,

XML DOM interfaces, and XML registry modification methods disclosed by Armstrong (Armstrong, p. 7, par. 2-3), to provide a more flexible and extensible programming interface for modifying XML documents using object-oriented programming constructs.

Regarding dependent claim 8, Skonnard teaches a series of functions for setting and returning a namespace association of a schema file to a markup applied to the document (p. 7; p. 11-12, Fig. 9 and 10), and teaches how to programmatically set a property in Fig. 10, thus creating the association, therefore it is implied in Skonnard that a new namespace is added to a collection, and a path and URI are passed to the schema library as parameters of the method object as shown at p. 7, par. 6-7 in the use of a standard namespace declaration to set the property.

Regarding dependent claim 9, Skonnard teaches a namespace manager class, which can be used to manage a stack of namespace bindings and their scope (p. 8, last paragraph), which allows registering namespaces and manifests in the schema library.

Regarding dependent claims 10-12, Skonnard teaches XPath and XSLT extension functions for accessing individual markup language resources from a collection of ML resources using a numerical index or keyword, where the index is passed as a parameter with the method property, controlling an alias name associated with a specified namespace identified in the library, and passing a pointer to the specified document at Fig. 12, "Other XPath Extension Functions", and Figs. 8 and 9.

Regarding dependent claims 13-15, Skonnard teaches a series of functions for setting and returning a namespace association of a schema file to a markup applied to the document (p. 7; p. 11-12, Fig. 9 and 10). Skonnard teaches an interactive XPath

expression builder to associate a schema object with the input document to execute a transformation (p. 7, "XPath and Type Information"; especially p. 7, par. 8). Specifically, Skonnard teaches a series of functions for setting and returning a namespace association of a schema file to a markup applied to the document (p. 11-12, Fig. 9 and 10), and teaches how to programmatically set a property in Fig. 10, thus creating the association, therefore, it is implied in the disclosure of Skonnard that the function setting the association property allows removal of an association since the property may be reset, resulting in the removal of the association and receiving a return value from the schema library responsive to removal of the association, as depicted in Figs. 9 and 10.

Regarding dependent claims 16 and 17, Skonnard teaches XPath and XSLT extension functions for accessing individual transformations from a collection using a numerical index or keyword, where the index is passed as a parameter with the method property, and controlling an alias name associated with a specified namespace identified in the library at Fig. 12, "Other XPath Extension Functions", and Figs. 8 and 9.

Regarding dependent claim 18, Skonnard teaches an interactive XPath expression builder to associate a schema object with the input document to execute an XSLT transformation (p. 7, "XPath and Type Information"; especially p. 7, par. 8). Specifically, Skonnard teaches a series of functions for setting and returning a namespace association of a schema file to a markup applied to the document (p. 11-12, Fig. 9 and 10), and teaches how to programmatically set a property in Fig. 10, thus creating the association, therefore, it is implied in the disclosure of Skonnard that the function setting the association property allows removal of an association since the

property may be reset, resulting in the removal of the association and receiving a return value from the schema library responsive to removal of the association, as depicted in Figs. 9 and 10.

Regarding independent claim 19, which cites *a computer-readable storage medium having computer-executable instructions for modifying resources of a Markup Language (ML) schema library*, Skonnard teaches a Schema Object Model (SOM) which is loaded into a programmable object and associated with a namespace URI (p. 3, par. 1);. Skonnard teaches an API interface for allowing a user to select resources in a ML schema library (p. 4, par. 1; p. 2-3, "A New XML Schema API: SOM").

While Skonnard teaches a method to access a schema library, and calling and passing the root and node schema object properties of the schema library (p. 5-6), Skonnard does not explicitly teach *the instructions comprising: receiving an object-oriented message call on the ML schema library, wherein the object-oriented message call is configured to modify a ML schema file of the ML schema library,* since Skonnard teaches accessing properties of the schema library and that the user can drill down into a particular schema item using the appropriate interfaces (p. 6, par. 1-3), but does not explicitly teach modifying the schema file. However, Armstrong teaches several interfaces for passing object properties between Java, an object-oriented programming language, and XML.

Armstrong discloses the Java API for XML Processing, (JAXP), an object oriented API to modify an XML schema file (p. 6-8). Armstrong also teaches JAX_RPC, the Java API for XML-based RPC, which makes it possible for an application to define

its own XML schema and to use that schema to send XML documents and XML fragments, as well as using the XML Simple Object Access Protocol (SOAP) to make an object oriented message call (p. 16-17). Since Armstrong teaches passing and receiving object properties using SOAP, and RPC (p. 16-17, p. 329-366), for web components (p. 337-338), i.e., software objects associated with functionality identified in the schema, Armstrong teaches *receiving an object property associated with the object-oriented message call, wherein the object property is associated with a software object for modifying the functionality of the ML schema file of the ML schema library;*

Armstrong teaches, *in response to receiving the object-oriented message call and the object property, modifying the functionality of the ML schema file of the ML schema library; and associating the modified functionality of the ML schema file with a document to govern the application of ML elements on the document*, because Armstrong teaches JAXR (p. 329-366) to access, submit data to an XML registry, modify it, and remove it (p. 355-360).

Both Skonnard and Armstrong are directed toward API interfaces for linking object-oriented programming languages with XML schema. It would have been obvious and desirable to one of ordinary skill in the art at the time of the invention to combine the methods of accessing XML schema libraries using an object-oriented schema interface (SOM) disclosed by Skonnard (Skonnard, p. 3, par. 2-4) with the flexible APIs, XML DOM interfaces, and XML registry modification methods disclosed by Armstrong (Armstrong, p. 7, par. 2-3), to provide a more flexible and extensible programming interface for modifying XML documents using object-oriented programming constructs.

Regarding dependent claim 20, while Skonnard does not explicitly teach modifying the schema file, Armstrong teaches *that modifying the functionality of the ML schema file of the ML schema library causes at least one member of a group comprising: applying ML markup to a document, and removing ML markup from a document*, because Armstrong teaches JAXR (p. 329-366) to access, submit data to an XML registry, modify it, and remove it (p. 355-360).

Both Skonnard and Armstrong are directed toward API interfaces for linking object-oriented programming languages with XML schema. It would have been obvious and desirable to one of ordinary skill in the art at the time of the invention to combine the methods of accessing XML schema libraries using an object-oriented schema interface (SOM) disclosed by Skonnard (Skonnard, p. 3, par. 2-4) with the flexible APIs, XML DOM interfaces, and XML registry modification methods disclosed by Armstrong (Armstrong, p. 7, par. 2-3), to provide a more flexible and extensible programming interface for modifying XML documents using object-oriented programming constructs.

Regarding dependent claim 21, while Skonnard does not explicitly teach modifying the schema file, Skonnard teaches *an interactive XPath expression builder to associate a schema object with the input document to execute an XSLT transformation* (p. 7, "XPath and Type Information"; especially p. 7, par. 8).

Armstrong teaches JAXR (p. 329-366) to access, submit data to an XML registry, modify it, and remove it (p. 355-360). Armstrong also teaches executing XSLT transformations on the XML DOM (p. 81-82).

Both Skonnard and Armstrong are directed toward API interfaces for linking object-oriented programming languages with XML schema. It would have been obvious and desirable to one of ordinary skill in the art at the time of the invention to combine the methods of accessing XML schema libraries using an object-oriented schema interface (SOM) disclosed by Skonnard (Skonnard, p. 3, par. 2-4) with the flexible APIs, XML DOM interfaces, and XML registry modification methods disclosed by Armstrong (Armstrong, p. 7, par. 2-3), to provide a more flexible and extensible programming interface for modifying XML documents using object-oriented programming constructs.

Regarding dependent claim 22, while Skonnard does not explicitly teach modifying the schema file, Armstrong teaches *modifying the functionality of the ML schema file of the ML schema library causes at least one member of a group comprising: associating an ML based resource with ML markup applied to a document, and removing an association of an ML based resource with ML markup applied to a documen*t, because Armstrong discloses the Java API for XML Processing, (JAXP), an object oriented API to modify an XML schema file (p. 6-8). Armstrong also teaches JAX_RPC, the Java API for XML-based RPC, which makes it possible for an application to define its own XML schema and to use that schema to send XML documents and XML fragments, as well as using the XML Simple Object Access Protocol (SOAP) to make an object oriented message call (p. 16-17). Armstrong teaches passing and receiving object properties using SOAP, and RPC (p. 16-17, p. 329-366), for web components (p. 337-338), i.e., software objects associated with functionality identified in the schema.

Both Skonnard and Armstrong are directed toward API interfaces for linking object-oriented programming languages with XML schema. It would have been obvious and desirable to one of ordinary skill in the art at the time of the invention to combine the methods of accessing XML schema libraries using an object-oriented schema interface (SOM) disclosed by Skonnard (Skonnard, p. 3, par. 2-4) with the flexible APIs, XML DOM interfaces, and XML registry modification methods disclosed by Armstrong (Armstrong, p. 7, par. 2-3), to provide a more flexible and extensible programming interface for modifying XML documents using object-oriented programming constructs.

Regarding independent claim 23, claim 23 is directed to the computer system for implementing the computer-readable storage medium having computer-executable instructions as claimed in claim 19, and is rejected along the same rationale.

Regarding dependent claim 24, claim 24 contains limitations substantially similar to the limitations of dependent claim 20, and is rejected along the same rationale.

Response to Arguments

6. Applicant's arguments with respect to claims 7-24 have been considered but are moot in view of the new ground(s) of rejection. The new ground of rejection includes the Armstrong reference, which is relied upon to teach the newly claimed limitations, including *...wherein the object-oriented message call is configured to modify a ML schema file of the ML schema library* (Claim 7).

Conclusion

7. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Kuramitsu, et al., "Distributed object-oriented schema for XML-based Electronic Catalog Sharing Semantics among Businesses", Web Information Systems Engineering, 2000, published June 2000, IEEE, p. 87-96.

Ingersoll et al.	**U.S. Patent No.**	**7,047,488 B2**	**issued**	**May 2006**
Kauffman et al.	**U.S. Patent No.**	**5,787,413**	**issued**	**July 1998**

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Amelia Rutledge whose telephone number is 571-272-7508. The examiner can normally be reached on Monday - Friday 9:30 - 6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Doug Hutton can be reached on 571-272-4137. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AR

William L. Bashore

WILLIAM BASHORE
PRIMARY EXAMINER